Midterm 2 Study Guide

The test covers all the material discussed in chapters 6, 7, 8, and sections 9.1-9.3, and homeworks 5 through 8. The following set of questions is meant to help guide your study and is not meant to be exhaustive of all the possibilities.

1. Describe what the *universal Python program* does, and provide code for it.
2. Define the language YesOnString and show that it is recognizable (by writing a Python program that recognizes it).
3. Write a program that recognizes, but does not decide, the language of genetic strings containing "GAGA". (Can you generalize your proof to *any* decidable language?)
4. Give equivalent formulations to the phrase "problem A is (Turing) reducible to problem B".
5. Suppose that problem A Turing reduces to problem B. Which of the following are necessarily true?
   - If A is computable then B is computable.
   - If A is undecidable then B is undecidable.
   - If B also reduces to A then A and B must be the same problem.
   - If B reduces to C then A reduces to C.
   - If A reduces to C then B reduces to C.
   - If we have an *oracle* program for deciding A, we can use it to solve problem B.
   - If we have an *oracle* program for deciding B, we can use it to solve problem A.
   - Problem A is solvable.
   - Problem B is solvable.
   - Problem A is no harder to solve than problem B.
6. Prove the following reductions, using explicit Python code where needed.
   - HaltsOnEmpty reduces to HaltsOnAllNonempty.
   - HaltsOnAllNonempty reduces to HaltsOnEmpty.
   - HaltsOnEmpty reduces to YesOnString.
   - YesOnString reduces to HaltsOnEmpty.
7. Define what it means for a Python program to *halt* (page 126)
8. Define the computational problem Computes_F for a given computational problem F.

9. True or False:

   - If F is computable then Computes_F is also computable.
   - If F is uncomputable then Computes_F is computable.
   - If F is computable then Computes_F is uncomputable.
   - If F is uncomputable then Computes_F is also uncomputable.

10. Give an example of a question about programs that is actually computable.

11. Describe how, given a computation tree for a non-deterministic program, we can determine the output of the computation (page 150).

12. Given a non-deterministic Turing Machine and an input string, draw the corresponding computation tree and determine the outcome of the computation. (Like figure 8.10 but showing the complete computation history)

13. Describe precisely how a non-deterministic Turing Machine differs from a deterministic Turing Machine.

14. Let L be an undecidable language. Prove that if L is recognizable then its complement $\bar{L}$ is unrecognizable.

15. Classify the following languages according to whether they are: decidable, recognizable but undecidable, corecognizable but undecidable, or neither recognizable nor corecognizable (nasty).

    - YesOnString
    - NotYesOnString
    - containsGAGA
    - HaltsOnEmpty
    - Complement of HaltsOnEmpty
    - Valid Python programs
    - Python programs that do not halt on empty

16. Give a precise definition of a *deterministic finite automaton (DFA)*.

17. Can a DFA loop on a given input? Explain.

18. Give a precise definition of a *non-deterministic finite automaton (NFA)*. What is different about a *strict* NFA?

19. Given an NFA, draw an equivalent DFA (like Figure 9.7).

20. Provide as many examples as you can of languages that can be decided by an NFA but not by a DFA.